General Approximate Cross Validation for Model Selection: Supervised, Semi-supervised and Pairwise Learning

Bowei Zhu
bowei.zhu@ruc.edu.cn
Renmin University of China
Beijing, China

Yong Liu*
liuyonggsai@ruc.edu.cn
Renmin University of China
Beijing, China

ABSTRACT

Cross-validation (CV) is a ubiquitous model-agnostic tool for assessing the error of machine learning. However, it has high complexity due to the requirement of multiple times of learner training especially in multimedia tasks with huge amounts of data. In this paper, we provide a unified framework to approximate the CV error for various common multimedia tasks such as supervised, semi-supervised and pairwise learning which requires training only once. Moreover, we study the theoretical performance of the proposed approximate CV and provide an explicit finite-sample error bound. Experimental results on several datasets demonstrate that our approximate CV has no statistical discrepancy from the original CV, but can significantly improve the efficiency, which is a great advantage in model selection.

CCS CONCEPTS

• **Computing methodologies** → **Semi-supervised learning settings**; **Cross-validation**; **Neural networks**.

KEYWORDS

machine learning, cross-validation,semi-supervised learning, pairwise learning

ACM Reference Format:
Bowei Zhu and Yong Liu. 2021. General Approximate Cross Validation for Model Selection: Supervised, Semi-supervised and Pairwise Learning. In *Proceedings of the 29th ACM Int'l Conference on Multimedia (MM '21), Oct. 20–24, 2021, Virtual Event, China.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3474085.3475649

1 INTRODUCTION

Cross-validation (CV) is the most popular model-agnostic tool for selecting the optimal model [23], which has been widely used in different kinds of multimedia tasks. Discussions and theoretical studies on the general t-fold CV can be found in [7, 27, 28, 38] and the references therein. However, CV is often prohibitively slow for modern, massive datasets, as they require running a learning algorithm on many slightly different datasets [33].

To accelerate the computation of CV, a number of papers have addressed for the special case of leave-one-out cross-validation in supervised learning: such as [8, 10, 32] for least square SVM (LSSVM), [9] for sparse least square SVM, [13, 14] for kernel ridge regression (KRR), [5, 14, 25] for general linear models, [35] for Bayesian linear regression, etc. Several authors have also considered the general t-fold CV. Based on the fact that LSSVM and KRR have closed-form solutions, [3, 37] applied the matrix inverse formula to develop a new method to expedite the t-fold CV process. [16, 17] used the infinitesimal jackknife to accelerate the t-fold CV and bootstrap. [30, 31, 33, 34] applied the Bouligand influence function to approximate the CV for kernel-based algorithms.

Although there are some studies on improving the efficiency of CV, most of the existing approaches employ only for one learning algorithm (or a few special learning algorithms) in supervised learning. Moreover, their analysis tend to lack the theoretical analysis. In this paper, we present a general framework to approximate CV for supervised, semi-supervised and pairwise learning, and further provide a theoretical guarantee for its performance. Specifically, we first provide a strategy to approximate the CV error via an approximation of the Taylor expansion, which requires one round of training. Then, we provide an explicit finite-sample error bound under some mild assumptions.

Furthermore, we give some specific examples for linear, nonlinear and pairwise cases. Experiments with CV method demonstrate that our framework is accurate. Further validation on different datasets prove that our approximate CV can not only provide the comparable results to the original one, but also significantly improve the efficiency. Our approximate CV shows good performance in model selection in areas including supervised learning, semi-supervised learning, contrastive learning, etc. The major contributions of the paper include:

(1) **A General Framework.** We propose a general framework for the approximation of the general t-fold CV for supervised learning, semi-supervised learning and pairwise learning, as well as for both linear and non-linear cases. In contrast, existing approximate CV approaches, such as [8, 10, 13, 14, 25, 35], only focus on leave-one-out CV, while [24, 30, 33, 34] focus on the kernel-based algorithms. [15, 18, 26, 39, 41–43] provides approximate CV for groups, but can only be applied to supervised learning. Besides for the supervised case, our method can be also used for semi-supervised learning and pairwise learning. Existing studies [11, 19, 44] on pairwise loss combine points into pairs and then bring the pairs as a whole into the approximation method given by [25, 26]. Our research gives a framework that can be used for pairwise learning

*Corresponding author

MM '21, October 20–24, 2021, Virtual Event, China.

ACM ISBN 978-1-4503-8651-7/21/10…$15.00
https://doi.org/10.1145/3474085.3475649

without prior combination, which is natural and can save memory. To the best of our knowledge, an approximation on general t-fold CV for both supervised and semi-supervised learning has never been given before.

(2) **Technological Novelty.** Most existing approximate methods such as [10, 17, 22, 25, 30, 33–35], only for supervised case. However, in our general framework, a pairwise loss depends on the unlabeled data is considered (see in Eq.(2)), so the technological skills of [10, 17, 25, 30, 33–35, 40] cannot be directly used for our general framework. To address this problem, we present a new strategy to derive the first and second derivatives with respect to the slight perturbation of the training set (see Theorem 3.1 and Theorem 4.1).

(3) **Algorithm Contribution.** We offer an efficient calculation method to compute the approximate cross-validation (see in Section 5.3). The method [1] computes for each training point with a fixed test point. But we calculate our approximate cross-validation for a certain set which is removed with each test data. We further extend this efficient calculation method to pairwise learning as well as non-convex and non-convergent problems.

(4) **Theoretical Contribution.** Based on the second derivative (Theorem 4.1) and the Taylor remainder [29], we present an explicit finite-sample error bound of our approximate CV (Theorem 4.5). In contrast, most existing methods, including [2–4, 20, 21, 25, 30, 34, 37, 45, 46], only provide supporting experiments and little theory.

The rest of the paper is organized as follows. We give a unified framework and associated notation in Section 2. In Section 3, we introduce an elegant strategy for approximate the CV. In Section 4, we present a theoretical guarantee of our approximate CV. In Section 5, we give some extensions for non-convexity and non-convergence. In Section 6, we provide some specific examples as well as the acceleration technology used in the approximate calculation. In Section 7, we empirically analyze the performance of our proposed approximate and give examples for model selection. We end in Section 8 with a conclusion. All the proofs are given in the supplementary material.

2 A UNIFIED FRAMEWORK

2.1 A General Form

Let a number of samples be defined as $\mathcal{L} = \{\mathbf{z}_i = (\mathbf{x}_i, y_i), i = 1, \ldots, n\}$, $\mathcal{S} \subseteq \mathcal{L}$, $|\mathcal{S}| = s$, $\mathcal{P} \subseteq \mathcal{L}$, $|\mathcal{P}| = p$. There are an input space $\mathcal{X}$ and an output space $\mathcal{Y}$. For input $x \in \mathcal{X}$ and label $y \in \mathcal{Y}$, parameter $\boldsymbol{\theta} \in \Theta$, our goal is to minimize the structural risk. In this paper, we consider the following general learning freamework:

$$\begin{aligned}\hat{\boldsymbol{\theta}}(\mathbf{1}) = \underset{\boldsymbol{\theta}\in\Theta}{argmin}\, F(\mathbf{1};\boldsymbol{\theta}) := \frac{\gamma_S}{s}\sum_{i=1}^{s}\ell_s(\mathbf{z}_i;\boldsymbol{\theta}) \\ + \frac{\gamma_P}{p^2}\sum_{i,j=1}^{p}\ell_p(\mathbf{z}_i,\mathbf{z}_j;\boldsymbol{\theta}) + \gamma_I R(\boldsymbol{\theta}),\end{aligned} \tag{1}$$

where ℓ_s is the loss for a single point function on set $\mathcal{S}$, ℓ_p is a pairwise loss function on set $\mathcal{P}$ and $R(\boldsymbol{\theta})$ is the regularization term. γ_S, γ_P and γ_I are three regularization parameters to trade off the balance among the pointwise loss, pairwise loss and regularization terms. The all-ones vector $\mathbf{1}$ in $\hat{\boldsymbol{\theta}}(\mathbf{1})$ denotes that the initial training points all have uniform sample weights of one. In this paper, we assume that ℓ_p is symmetrical, i.e., $\ell_p(z, z'; \boldsymbol{\theta}) = \ell_p(z', z; \boldsymbol{\theta})$.

Let $\mathcal{W} \subset \mathcal{S} \bigcap \mathcal{P}$ be a subset of training points removed from the whole set of points. $|\mathcal{W}| = v$ and $\mathbf{w} \in \{0,1\}^n$ be a vector of sample weights with $w_i = 0$ if $\mathbf{z}_i \in \mathcal{W}$, otherwise 1. Denote as $\hat{\boldsymbol{\theta}}(\mathbf{w})$ the model learned on all of the data excluding $\mathcal{W}$:

$$\begin{aligned}\hat{\boldsymbol{\theta}}(\mathbf{w}) = \underset{\boldsymbol{\theta}\in\Theta}{argmin}\, F(\mathbf{w};\boldsymbol{\theta}) := \frac{\gamma_S}{s-v}\sum_{i=1}^{s} w_i\ell_s(\mathbf{z}_i;\boldsymbol{\theta}) + \\ \frac{\gamma_P}{(p-v)^2}\sum_{i,j=1}^{p} w_i w_j \ell_p(\mathbf{z}_i,\mathbf{z}_j;\boldsymbol{\theta}) + \gamma_I R(\boldsymbol{\theta}).\end{aligned} \tag{2}$$

We usually need to choose the appropriate evaluation function for cross-validation. In this paper, we use $\ell_e(\boldsymbol{\theta})$ to denote the evaluation function, which can be either a pointwise loss or a pairwise loss.

2.2 Examples

In this subsection, we provide some specific forms and evaluation functions under supervised learning, semi-supervised learning and pairwise learning, which is the basis for our approximation method.

2.2.1 SUPERVISED LEARNING. When γ_P is set to 0, our formula can be reduced to the form of supervised learning, which means only supervised samples are considered. In this condition, the structural risk is shown below:

$$\hat{\boldsymbol{\theta}}(\mathbf{1}) = \underset{\boldsymbol{\theta}\in\Theta}{argmin}\, F(\mathbf{1};\boldsymbol{\theta}) := \frac{1}{s}\sum_{i=1}^{s}\ell_s(\mathbf{z}_i;\boldsymbol{\theta}) + \gamma_I R(\boldsymbol{\theta}), \tag{3}$$

where γ_S in Eq.(1) may wish to be set to 1 to simplify the calculation, and $\mathcal{S}$ is the whole train dataset.

Let $\{\mathcal{S}_i\}_{i=1}^{t}$ be random equipartitions of $\mathcal{S}$ into t parts, and $\pi_i := \{j | \mathbf{z}_j = (\mathbf{x}_j, y_j) \in \mathcal{S}_i\}$. For simplicity, assume that s mod t, hence, $|\mathcal{S}_i| = \frac{s}{t} =: m$ and $|\pi_i| = m$. Let $\hat{\boldsymbol{\theta}}(\mathbf{w}_{\setminus\mathcal{S}_i})$ be the hypothesis learned on all of the data excluding $\mathcal{S}_i$, where $[\mathbf{w}_{\setminus\mathcal{S}_i}]_k = 0$, if $(\mathbf{x}_k, y_k) \in \mathcal{S}_i$, otherwise 1, $k \in \{1, \ldots, s\}$. Then, the general t-fold CV (t-CV) can be written as:

$$t\text{-CV} := \frac{1}{s}\sum_{i=1}^{t}\sum_{\mathbf{z}_j\in\mathcal{S}_i}\ell_e\left(\mathbf{z}_j;\hat{\boldsymbol{\theta}}(\mathbf{w}_{\setminus\mathcal{S}_i})\right). \tag{4}$$

When $t = s$, the s-CV is a special case of t-fold CV, called the leave-one-out CV error (LOO):

$$\text{LOO} := \frac{1}{s}\sum_{i=1}^{s}\ell_e\left(\mathbf{z}_i;\hat{\boldsymbol{\theta}}(\mathbf{w}_{\setminus i})\right). \tag{5}$$

2.2.2 SEMI-SUPERVISED LEARNING. Our framework can be used to solve semi-supervised problems, when pairwise loss uses only the feature part of the point z. In this condition, $\mathcal{S}$ can be considered as a small number of labeled samples, and $\mathcal{P} - \mathcal{S}$ is a large number of unlabeled samples. Similarly, the value of γ_S can be set

to 1. The structural risk at this point is as follow:

$$\begin{aligned}\hat{\boldsymbol{\theta}}(\mathbf{1}) = \underset{\boldsymbol{\theta}\in\Theta}{argmin}\, F(\mathbf{1};\boldsymbol{\theta}) := \frac{1}{s}\sum_{i=1}^{s}\ell_s(\mathbf{z}_i;\boldsymbol{\theta}) \\ + \frac{\gamma_P}{p^2}\sum_{i,j=1}^{p}\ell_p(\mathbf{x}_i,\mathbf{x}_j;\boldsymbol{\theta}) + \gamma_I R(\boldsymbol{\theta}).\end{aligned} \tag{6}$$

Since the labeled data must be used when evaluating model performance in cross-validation, the evaluation function, t-CV and LOO for semi-supervised learning is the same as Eq.(4).

2.2.3 PAIRWISE LEARNING. When the pointwise term is thrown away, the whole structural risk can be regraded as pairwise learning loss. So this framework can be applied in pairwise learning such as constractive learning for images. In this case, we need to minimize the following function:

$$\hat{\boldsymbol{\theta}}(\mathbf{1}) = \underset{\boldsymbol{\theta}\in\Theta}{argmin}\, F(\mathbf{1};\boldsymbol{\theta}) := \frac{1}{p^2}\sum_{i,j=1}^{p}\ell_p(\mathbf{z}_i,\mathbf{z}_j;\boldsymbol{\theta}) + \gamma_I R(\boldsymbol{\theta}), \tag{7}$$

where the pointwise term was thrown away and γ_P is set to 1 to simplify the calculation. Similarly, let $\{\mathcal{P}_i\}_{i=1}^{t}$ be random equipartitions of $\mathcal{P}$ into t parts, $|\mathcal{P}_i| = \frac{p}{t} =: m$, the general t-fold CV (t-CV) can be written as:

$$t\text{-CV} := \frac{1}{mp}\sum_{i=1}^{t}\sum_{(\mathbf{z}_j,\mathbf{z}_k)\in\mathcal{P}_i}\ell_e\left(\mathbf{z}_j,\mathbf{z}_k;\hat{\boldsymbol{\theta}}(\mathbf{w}_{\setminus\mathcal{P}_i})\right). \tag{8}$$

The leave-one-out CV error (LOO) for pairwise learning can be simplified as

$$\text{LOO} := \frac{1}{p^2}\sum_{(\mathbf{z}_i,\mathbf{z}_j)\in\mathcal{P}}\ell_e\left(\mathbf{z}_i,\mathbf{z}_j;\hat{\boldsymbol{\theta}}(\mathbf{w}_{\setminus\{\mathbf{z}_i,\mathbf{z}_j\}})\right), \tag{9}$$

where $\hat{\boldsymbol{\theta}}(\mathbf{w}_{\setminus\{\mathbf{z}_i,\mathbf{z}_j\}})$ denotes the $\boldsymbol{\theta}$ obtained by training after removing points $\{\mathbf{z}_i,\mathbf{z}_j\}$.

From the above definition of t-CV, one can see that we require the learning algorithm to be run t times to obtain t-CV, which is time-consuming.

3 FAST APROXIMATE CROSS-VALIDATION

In this section, we provide the fast cross-validation approximation algorithm, and give its specific form under supervised, semi-supervised and pairwise learning.

3.1 Talor Expansion Approximation

Note that $\mathcal{W}$ is a small subset of $\mathcal{L}$, so $\hat{\boldsymbol{\theta}}(\mathbf{w}_{\setminus\mathcal{W}})$ can be considered a slight disturbance of $\hat{\boldsymbol{\theta}}(\mathbf{1})$. Therefore, the Taylor expansion approximation [29] at the point $\hat{\boldsymbol{\theta}}(\mathbf{1})$ is an effective approximation of $\hat{\boldsymbol{\theta}}(\mathbf{w}_{\setminus\mathcal{W}})$. Let $\boldsymbol{\delta}_{\setminus\mathcal{W}} = \mathbf{w}_{\setminus\mathcal{W}} - \mathbf{1}$ be the sample discrepancy between $\mathbf{w}_{\setminus\mathcal{W}}$ and $\mathbf{1}$. Considering a certain evaluation function with the Taylor expansion approximation, for the specific data (this may be a point data or a point-to-point data) we need to evaluate, we have

$$\begin{aligned}&\ell_e(\hat{\boldsymbol{\theta}}(\mathbf{w}_{\setminus\mathcal{W}})) \approx \tilde{\ell}_e(\hat{\boldsymbol{\theta}}(\mathbf{w}_{\setminus\mathcal{W}})) \\ :=&\ell_e(\hat{\boldsymbol{\theta}}(\mathbf{1})) + \left[\nabla_{\mathbf{w}}\ell_e(\hat{\boldsymbol{\theta}}(\mathbf{1}))^{\mathrm{T}}\middle|\mathbf{w}=1\right]\boldsymbol{\delta}_{\setminus\mathcal{W}} \\ =&\ell_e(\hat{\boldsymbol{\theta}}(\mathbf{1})) + \nabla_{\boldsymbol{\theta}}\ell_e(\hat{\boldsymbol{\theta}}(\mathbf{1}))^{\mathrm{T}}\left[\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w})}{d\mathbf{w}^{\mathrm{T}}}\Big|_{\mathbf{w}=1}\right]\boldsymbol{\delta}_{\setminus\mathcal{W}},\end{aligned} \tag{10}$$

where ℓ_e is the evaluation function, such as square loss, Logistic loss, pairwise distance etc. Here we use the chain rule of derivation. We can see that if $\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w})}{d\mathbf{w}^{\mathrm{T}}}$ is given, we only need to train the algorithm **once** on the full dataset $\mathcal{L}$ to obtain $\hat{\boldsymbol{\theta}}(\mathbf{1})$.

3.2 General Calculation Form

First, we show how to calculate $\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w})}{d\mathbf{w}^{\mathrm{T}}}$ for the general form.

THEOREM 3.1. *Let $\mathbf{H}_{\mathbf{w}} = \nabla^2_{\boldsymbol{\theta}}F(\mathbf{w};\hat{\boldsymbol{\theta}}(\mathbf{w}))$ be the second derivative of the object $F(\mathbf{w};\boldsymbol{\theta})$ (defined in Eq.(2)) at $\hat{\boldsymbol{\theta}}(\mathbf{w})$, and assume it exists. Then, we have*

$$\begin{aligned}\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w})}{dw_i} = -\mathbf{H}_{\mathbf{w}}^{-1}\Big(&\frac{\gamma_S}{s-v}\nabla_{\boldsymbol{\theta}}\ell_s(\mathbf{z}_i;\hat{\boldsymbol{\theta}}(\mathbf{w})) + \\ &\frac{2\gamma_P}{(p-v)^2}\sum_{j=1}w_j\nabla_{\boldsymbol{\theta}}\ell_p(\mathbf{z}_i,\mathbf{z}_j;\hat{\boldsymbol{\theta}}(\mathbf{w}))\Big).\end{aligned}$$

REMARK 1. *Theorem 3.1 gives a unified framework to calculate $\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w})}{d\mathbf{w}^{\mathrm{T}}}$ for supervised, semi-supervised and pairwise learning. The approximate calculation of $\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w})}{d\mathbf{w}^{\mathrm{T}}}$ in [25, 30, 33, 34] can be considered as special cases of our framework when only considering the supervised samples. And this can be used in pairwise learning with γ_S equaled to 0.*

Let $\mathbf{g}(\mathbf{w}_{\setminus\mathcal{W}}) = \frac{\gamma_S}{s-v}\sum_{i=1}^{s}(w_i-1)\nabla_{\boldsymbol{\theta}}\ell_s(\mathbf{z}_i;\hat{\boldsymbol{\theta}}(\mathbf{1})) + \frac{2\gamma_P}{(p-v)^2}$ $\sum_{i,j=1}^{p}(w_i-1)w_j\nabla_{\boldsymbol{\theta}}\ell_p(\mathbf{z}_i,\mathbf{z}_j;\hat{\boldsymbol{\theta}}(\mathbf{1}))$, and $\mathbf{H} = \mathbf{H}_{\mathbf{1}} = \nabla^2_{\boldsymbol{\theta}}F(\mathbf{1};\hat{\boldsymbol{\theta}}(\mathbf{1}))$. From Theorem 3.1, we can obtain that

$$\left[\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w})}{d\mathbf{w}^{\mathrm{T}}}\Big|_{\mathbf{w}=1}\right]\boldsymbol{\delta}_{\setminus\mathcal{W}} = -\mathbf{H}^{-1}\mathbf{g}(\mathbf{w}_{\setminus\mathcal{W}}).$$

Thus, the approximation of the evaluation function is

$$\ell_e(\hat{\boldsymbol{\theta}}(\mathbf{w}_{\setminus\mathcal{W}})) \approx \ell_e(\hat{\boldsymbol{\theta}}(\mathbf{1})) - \nabla_{\boldsymbol{\theta}}\ell_e(\hat{\boldsymbol{\theta}}(\mathbf{1}))^{\mathrm{T}}\mathbf{H}^{-1}\mathbf{g}(\mathbf{w}_{\setminus\mathcal{W}}). \tag{11}$$

This formula gives us a simple way to calculate the evaluation function. We can calculate the scores of the model under cross-validation by training the full dataset only **once**, which can further help us to filter the best models and hyperparameters.

3.3 Specific calculation formula

3.3.1 SUPERVISED LEARNING. When γ_P is set to 0, this approximation method can be reduced to calculate supervised learning. According to Eq.(3), Eq.(4) and Eq.(11), the approximate t-CV can be written as

$$\begin{aligned}t\text{-ACV} :=&\frac{1}{s}\sum_{i=1}^{t}\sum_{\mathbf{z}_j\in\mathcal{S}_i}\Big(\ell_e(\mathbf{z}_j;\hat{\boldsymbol{\theta}}(\mathbf{1})) \\ &-\nabla_{\boldsymbol{\theta}}\ell_e(\mathbf{z}_j;\hat{\boldsymbol{\theta}}(\mathbf{1}))^{\mathrm{T}}\mathbf{H}^{-1}\mathbf{g}(\mathbf{w}_{\setminus\mathcal{S}_i})\Big),\end{aligned} \tag{12}$$

where $\mathbf{g}(\mathbf{w}_{\setminus\mathcal{S}_i}) = \frac{1}{s-m}\sum_{i=1}^{s}(w_i-1)\nabla_{\boldsymbol{\theta}}\ell_s(\mathbf{z}_i;\hat{\boldsymbol{\theta}}(\mathbf{1}))$, and $\mathbf{H} = \mathbf{H_1} = \nabla^2_{\boldsymbol{\theta}}F(\mathbf{1};\hat{\boldsymbol{\theta}}(\mathbf{1}))$.

The approximate LOO is a special case of the approximate t-CV with $t = s$, which can be represented as

$$\begin{aligned}\text{ALOO} :=& \frac{1}{s}\sum_{i=1}^{s}\Big(\ell_e(\mathbf{z}_i;\hat{\boldsymbol{\theta}}(\mathbf{1})) \\ &-\nabla_{\boldsymbol{\theta}}\ell_e(\mathbf{z}_i;\hat{\boldsymbol{\theta}}(\mathbf{1}))^{\mathrm{T}}\mathbf{H}^{-1}\mathbf{g}(\mathbf{w}_{\setminus i})\Big),\end{aligned} \tag{13}$$

where $\mathbf{g}(\mathbf{w}_{\setminus i}) = -\frac{1}{s}\nabla_{\boldsymbol{\theta}}\ell_s(\mathbf{z}_i;\hat{\boldsymbol{\theta}}(\mathbf{1}))$, and $\mathbf{H} = \mathbf{H_1} = \nabla^2_{\boldsymbol{\theta}}F(\mathbf{1};\hat{\boldsymbol{\theta}}(\mathbf{1}))$.

3.3.2 SEMI-SUPERVISED LEARNING. Combining Eq.(3), Eq.(4) and Eq.(11), we get the approximate t-CV which can be written as

$$\begin{aligned}t\text{-ACV} :=& \frac{1}{s}\sum_{i=1}^{t}\sum_{(\mathbf{x}_j,y_j)\in\mathcal{S}_i}\Big(\ell_e(\mathbf{z}_j;\hat{\boldsymbol{\theta}}(\mathbf{1})) \\ &-\nabla_{\boldsymbol{\theta}}\ell_e(\mathbf{z}_j;\hat{\boldsymbol{\theta}}(\mathbf{1}))^{\mathrm{T}}\mathbf{H}^{-1}\mathbf{g}(\mathbf{w}_{\setminus\mathcal{S}_i})\Big),\end{aligned} \tag{14}$$

where $\mathbf{g}(\mathbf{w}_{\setminus\mathcal{S}_i}) = \frac{1}{s-m}\sum_{i=1}^{s}(w_i-1)\nabla_{\boldsymbol{\theta}}\ell_s(\mathbf{z}_i;\hat{\boldsymbol{\theta}}(\mathbf{1})) + \frac{2\gamma_P}{(p-m)^2}$ $\sum_{i,j=1}^{p}(w_i-1)w_j\nabla_{\boldsymbol{\theta}}\ell_p(\mathbf{x}_i,\mathbf{x}_j;\hat{\boldsymbol{\theta}}(\mathbf{1}))$ and $\mathbf{H} = \mathbf{H_1} = \nabla^2_{\boldsymbol{\theta}}F(\mathbf{1};\hat{\boldsymbol{\theta}}(\mathbf{1}))$. Comparing Eq.(12) with Eq.(14), we can find that the approximate cross-validation algorithm for semi-supervised learning has one more term $\frac{2\gamma_P}{(p-m)^2}\sum_{i,j=1}^{p}(w_i-1)w_j\nabla_{\boldsymbol{\theta}}\ell_p(\mathbf{x}_i,\mathbf{x}_j;\hat{\boldsymbol{\theta}}(\mathbf{1}))$, which indicates the contribution of unlabeled samples.

The approximate LOO is a special case of the approximate t-CV with $t = s$, which can be represent as

$$\begin{aligned}\text{ALOO} :=& \frac{1}{s}\sum_{i=1}^{l}\Big(\ell_e(\mathbf{z}_i;\hat{\boldsymbol{\theta}}(\mathbf{1})) \\ &-\nabla_{\boldsymbol{\theta}}\ell_e(\mathbf{z}_i;\hat{\boldsymbol{\theta}}(\mathbf{1}))^{\mathrm{T}}\mathbf{H}^{-1}\mathbf{g}_{\setminus i}\Big),\end{aligned} \tag{15}$$

where $\mathbf{g}_{\setminus i} = -\frac{1}{s-1}\nabla_{\boldsymbol{\theta}}\ell_s(\mathbf{z}_i;\hat{\boldsymbol{\theta}}(\mathbf{1})) - \frac{2\gamma_P}{(p-1)^2}\sum_{j=1}^{p}\nabla_{\boldsymbol{\theta}}\ell_p(\mathbf{x}_i,\mathbf{x}_j;\hat{\boldsymbol{\theta}}(\mathbf{1}))$.

Approximation algorithms for cross-validation and leave-one-out methods in semi-supervised learning in the general case are given by Eq. (14) and Eq. (15). Using this approximation, we can evaluate the model is by training it only once. To the best of our knowledge, the proof is first given for semi-supervised learning.

3.3.3 PAIRWISE LEARNING. When γ_S is set to 0, the approximation algorithm can be used for pairwise learning. According to Eq.(7), Eq.(8) and Eq.(11), the approximate t-CV can be written as

$$\begin{aligned}t\text{-ACV} :=& \frac{1}{mp}\sum_{i=1}^{t}\sum_{(\mathbf{z}_j,\mathbf{z}_k)\in\mathcal{P}_i}\Big(\ell_e(\mathbf{z}_j,\mathbf{z}_k;\hat{\boldsymbol{\theta}}(\mathbf{1})) \\ &-\nabla_{\boldsymbol{\theta}}\ell_e(\mathbf{z}_j,\mathbf{z}_k;\hat{\boldsymbol{\theta}}(\mathbf{1}))^{\mathrm{T}}\mathbf{H}^{-1}\mathbf{g}(\mathbf{w}_{\setminus\mathcal{S}_i})\Big),\end{aligned} \tag{16}$$

where $\mathbf{g}(\mathbf{w}_{\setminus\mathcal{S}_i}) = \frac{2}{(p-m)^2}\sum_{i,j=1}^{p}(w_i-1)w_j\nabla_{\boldsymbol{\theta}}\ell_p(\mathbf{z}_i,\mathbf{z}_j;\hat{\boldsymbol{\theta}}(\mathbf{1}))$ and $\mathbf{H} = \mathbf{H_1} = \nabla^2_{\boldsymbol{\theta}}F(\mathbf{1};\hat{\boldsymbol{\theta}}(\mathbf{1}))$. Unlike Eq.(14), Eq.(16) is related to the derivative of a pair of points rather than a single point w.r.t. the loss.

Similarly, the approximate LOO is a special case of the approximate t-CV with $t = p$, which can be represent as

$$\begin{aligned}\text{ALOO} :=& \frac{1}{p^2}\sum_{(\mathbf{z}_i,\mathbf{z}_j)\in\mathcal{P}}\Big(\ell_e(\mathbf{z}_i,\mathbf{z}_j;\hat{\boldsymbol{\theta}}(\mathbf{1})) \\ &-\nabla_{\boldsymbol{\theta}}\ell_e(\mathbf{z}_i,\mathbf{z}_j;\hat{\boldsymbol{\theta}}(\mathbf{1}))^{\mathrm{T}}\mathbf{H}^{-1}\mathbf{g}_{\setminus i}\Big),\end{aligned} \tag{17}$$

where $\mathbf{g}_{\setminus\{\mathbf{z}_i,\mathbf{z}_j\}} = -\frac{2}{(p-1)^2}\sum_{k=1}^{p}\nabla_{\boldsymbol{\theta}}\ell_p(\mathbf{z}_i,\mathbf{z}_k;\hat{\boldsymbol{\theta}}(\mathbf{1})) - \frac{2}{(p-1)^2}$ $\sum_{k=1}^{p}\nabla_{\boldsymbol{\theta}}\ell_p(\mathbf{z}_j,\mathbf{z}_k;\hat{\boldsymbol{\theta}}(\mathbf{1}))$.

Existing studies [19, 44] on pairwise loss combines points into pairs and then bring the pairs as a whole into the approximation method given by [25]. Our research gives a framework that can be used for pairwise learning without prior combination. Approximation algorithms for cross-validation and leave-one-out methods in pairwise learning in the general case are given by Eq. (16) and Eq. (17), which have never been given before.

3.4 Complexity Analysis

Traditional t-fold CV is complex and need to train the algorithm t times, if we assume that it takes $O(T)$ time to train the model once, the time complexity of general t-fold CV will reach $O(tT)$. When we analyses our approximate CV method, we assume that it spends $O(T')$ time to train the model once using the whole data. Since the time complexity of our method is almost the same as the traditional method to train the model once, which means that $T \approx T'$. The time taken to compute the approximation is much less than the time taken to train the model. In consequence, our method is nearly t times faster especially when t is large.

4 ERROR ANALYZATION

4.1 Unified Error Analysis

From Taylor expansion [29], we know that

$$\ell_e(\hat{\boldsymbol{\theta}}(\mathbf{w}_{\setminus\mathcal{W}})) = \tilde{\ell}_e(\hat{\boldsymbol{\theta}}(\mathbf{w}_{\setminus\mathcal{W}})) + r(\hat{\boldsymbol{\theta}}(\mathbf{w}')),$$

where $r(\hat{\boldsymbol{\theta}}(\mathbf{w}')) = \boldsymbol{\delta}^{\mathrm{T}}_{\mathbf{w}_{\setminus\mathcal{W}}}\nabla^2_{\mathbf{w}}\ell(\hat{\boldsymbol{\theta}}(\mathbf{w}'))\boldsymbol{\delta}_{\mathbf{w}_{\setminus\mathcal{W}}}$ is the Lagrange form of the Taylor remainder, $\mathbf{w}' \in [\mathbf{w}_{\setminus\mathcal{W}}, \mathbf{1}]$, $\tilde{\ell}_e(\hat{\boldsymbol{\theta}}(\mathbf{w}_{\setminus\mathcal{W}}))$ is defined in Eq.(10). To derive the error bound, we need to require the Taylor remainder $r(\hat{\boldsymbol{\theta}}(\mathbf{w}'))$ to be bounded. Note that

$$\begin{aligned}&\left|r(\hat{\boldsymbol{\theta}}(\mathbf{w}'))\right| = \left|\boldsymbol{\delta}^{\mathrm{T}}_{\mathbf{w}_{\setminus\mathcal{W}}}\nabla^2_{\mathbf{w}}\ell_e(\hat{\boldsymbol{\theta}}(\mathbf{w}'))\boldsymbol{\delta}_{\mathbf{w}_{\setminus\mathcal{W}}}\right| \\ =&\left|\sum_{k,h=1}^{p}[\boldsymbol{\delta}_{\mathbf{w}_{\setminus\mathcal{W}}}]_k[\boldsymbol{\delta}_{\mathbf{w}_{\setminus\mathcal{W}}}]_h\left[\nabla_{\boldsymbol{\theta}}\ell_e(\hat{\boldsymbol{\theta}}(\mathbf{w}'))^{\mathrm{T}}\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w}')}{dw_k dw_h}\right.\right. \\ &\left.\left.+\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w}')}{dw_h}^{\mathrm{T}}\nabla^2_{\boldsymbol{\theta}}\ell_e(\hat{\boldsymbol{\theta}}(\mathbf{w}'))\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w}')}{dw_k}\right]\right| \\ \le&\sum_{k,h\in\pi_i}\left(\left\|\nabla_{\boldsymbol{\theta}}\ell_e(\hat{\boldsymbol{\theta}}(\mathbf{w}'))\right\|\left\|\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w}')}{dw_k dw_h}\right\|\right. \\ &\left.+\left\|\nabla^2_{\boldsymbol{\theta}}\ell_e(\hat{\boldsymbol{\theta}}(\mathbf{w}'))\right\|\left\|\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w}')}{dw_k}\right\|\left\|\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w}')}{dw_h}\right\|\right).\end{aligned} \tag{18}$$

Therefore, to derive the error bound, we should estimate $\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w})}{dw_i dw_j}$.

Theorem 4.1. *Let* $\mathbf{T}_{\mathbf{w}} = \nabla^3_{\boldsymbol{\theta}} F(\mathbf{w}; \hat{\boldsymbol{\theta}}(\mathbf{w}))$ *be the third derivatives of* $F(\mathbf{w}; \boldsymbol{\theta})$ *at* $\hat{\boldsymbol{\theta}}(\mathbf{w})$*, and assume that the second and third derivatives of* $F(\mathbf{w}; \hat{\boldsymbol{\theta}}(\mathbf{w}))$ *exist. Then, we have*

$$\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w})}{dw_j dw_i} = -\mathbf{H}_{\mathbf{w}}^{-1}\left[\mathbf{g}_p^{i,j} + \mathbf{G}^i \frac{d\hat{\boldsymbol{\theta}}(\mathbf{w})}{dw_j} + \mathbf{G}^j \frac{d\hat{\boldsymbol{\theta}}(\mathbf{w})}{dw_i} + \mathbf{T}_{\mathbf{w}} \frac{d\hat{\boldsymbol{\theta}}(\mathbf{w})}{dw_i} \frac{d\hat{\boldsymbol{\theta}}(\mathbf{w})}{dw_j}^{\mathrm{T}}\right],$$

where $\mathbf{g}_p^{i,j} = \frac{2\gamma_P}{(p-v)^2} \nabla_{\boldsymbol{\theta}} \ell_p(\mathbf{z}_i, \mathbf{z}_j; \hat{\boldsymbol{\theta}}(\mathbf{w}))$, $\mathbf{G}^i = \frac{\gamma_S}{s-v} \nabla^2_{\boldsymbol{\theta}} \ell_s(\mathbf{z}_i; \hat{\boldsymbol{\theta}}(\mathbf{w})) + \frac{2\gamma_P \sum_{k=1} w_k \nabla^2_{\boldsymbol{\theta}} \ell_p(\mathbf{z}_i, \mathbf{z}_k; \hat{\boldsymbol{\theta}}(\mathbf{w}))}{(p-v)^2}$.

Remark 2. *To obtain an error bound, we show how to estimate the second derivative of* $\boldsymbol{\theta}(\mathbf{w})$ *with respect to* $\mathbf{w}$ *in Theorem 4.1. However, most existing methods, such as [3, 13, 14, 25, 34, 37], etc, only derive the first derivative. The second derivative for a general framework is novel.*

Lemma 4.2. *If* $F(\mathbf{w}; \boldsymbol{\theta})$ *is an* σ*-strongly convex function with respect to* $\boldsymbol{\theta}$*, and* $\forall(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$, $\mathbf{x}' \in \mathcal{X}$, $\boldsymbol{\theta} \in \Theta$, $\|\nabla_{\boldsymbol{\theta}} \ell_s(z; \boldsymbol{\theta})\| \le c'_s$, $\|\nabla_{\boldsymbol{\theta}} \ell_p(z, z'; \boldsymbol{\theta})\| \le c'_p$*. Then, the following result holds:*

$$\left\|\frac{d\hat{\boldsymbol{\theta}}(\mathbf{w})}{dw_i}\right\| \le \frac{c'_s \gamma_S}{\sigma(s-v)} + \frac{2c'_p \gamma_P}{\sigma(p-v)} = O\left(\frac{1}{s-v} + \frac{1}{p-v}\right).$$

Remark 3. *Lemma 4.2 tells us that we can easily put an upper bound on the derivative of* θ *w.r.t.* $\mathbf{w}$*. The value of this upper bound also fits well with our experience that the larger the dataset, the smaller the influence of perturbations in the training set on the model. Meanwhile, the larger the perturbation, the bigger the influence on the model.*

It seems difficult for $F(\mathbf{w}; \boldsymbol{\theta})$ *to be* σ*-strong convex in practice. But we use only the property that* $\|\mathbf{H}_{\mathbf{w}}\|_2 = \|\nabla^2_{\boldsymbol{\theta}} F(\mathbf{w}; \boldsymbol{\theta})\|_2 \ge \sigma$ *during the whole proof. In fact, we can ensure that the above condition holds by add a damping term* $\boldsymbol{\lambda}$*. The details can be found in Section 4.2*

Lemma 4.3. *We have the same assumptions as Lemma 4.2, and* $\|\nabla^2_{\boldsymbol{\theta}} \ell_s(z; \boldsymbol{\theta})\| \le c''_s$, $\|\nabla^3_{\boldsymbol{\theta}} \ell_s(z; \boldsymbol{\theta})\| \le c'''_s$, $\|\nabla^2_{\boldsymbol{\theta}} \ell_p(z, z'; \boldsymbol{\theta})\| \le c''_p$, $\|\nabla^3_{\boldsymbol{\theta}} \ell_p(z, z'; \boldsymbol{\theta})\| \le c'''_p$ *and* $\|\nabla^3_{\boldsymbol{\theta}} R(\hat{\boldsymbol{\theta}}(\mathbf{w}))\| \le c'''_r$*. Thus, we have*

$$\left\|\frac{d\boldsymbol{\theta}(\mathbf{w})}{dw_j dw_i}\right\| = O\left(\frac{1}{(s-v)^2} + \frac{1}{(s-v)(p-v)} + \frac{1}{(p-v)^2}\right).$$

Remark 4. *Lemma 4.3 is similar to Lemma 4.2. When the second-order derivative and third-order derivative of functions* ℓ_s *and* ℓ_p *have upper bounds, the second-order derivative of* θ *also corresponds to an upper bound. Many popular loss function satisfy the above condition when the* $\mathbf{x}$ *is bounded, such as square loss, logistic loss and Huber loss.*

4.2 Error Bounds of Approximate CV

Next we give the error bounds of approximate CV in supervised learning, semi-supervised learning and pairwise learning.

It is easy to deduce the following conclusion for supervised and semi-supervised learning.

$$t\text{-ACV} - t\text{-CV} = \frac{1}{s}\sum_{i=1}^{s} r(\mathbf{z}_i; \hat{\boldsymbol{\theta}}(\mathbf{w}')). \tag{19}$$

4.2.1 SUPERVISED LEARNING. By Lemmas 4.2 and 4.3, when we set γ_P to 0, the error bound for supervised learning can be obtained:

Theorem 4.4. *Under the same assumptions as Lemma 4.3,* $\gamma_P = 0$*,*

$$|t\text{-ACV} - t\text{-CV}| = O\left(\frac{1}{(t-1)^2}\right),$$

where t *is the number of the fold of dataset,* m *is the size of each fold of dataset,* $t = s/m$*.*

When setting $t = s$, the error bound of the approximate LOO is $|\text{ALOO} - \text{LOO}| = O\left(\frac{1}{s^2}\right)$.

Remark 5. *Generally,* $s = n$ *when our framework is used in supervised learning. This theory tells us to use as larger dataset as possible and as many group as possible during cross-validation so that the approximation method can be more accurate.*

4.2.2 SEMI-SUPERVISED LEARNING. By Lemmas 4.2 and 4.3, the error bound can be obtained:

Theorem 4.5. *Under the same assumptions as Lemma 4.3,*

$$|t\text{-ACV} - t\text{-CV}| = O\left(\frac{m^2}{(s-m)^2} + \frac{m^2}{(s-m)(p-m)} + \frac{m^2}{(p-m)^2}\right),$$

where m *is the size of the fold of dataset,* $m = s/t$*.*

Similarly, the error bound of the approximate LOO is $|\text{ALOO} - \text{LOO}| = O\left(\frac{1}{s^2} + \frac{1}{sp} + \frac{1}{p^2}\right)$, when setting $t = s$.

Remark 6. *There are some work studies the approximations of the general* t*-fold CV, but most of them only provide supporting experiments and little theory. To the best of our knowledge, the only two results providing explicit finite-sample error bounds are [16, 17]. However, the proof of [16, 17] are very complex. In this paper, we directly bound the Taylor remainder, which is much simpler. Moreover, our result can also be applicable for semi-supervised learning but [16, 17] only for supervised one.*

4.2.3 PAIRWISE LEARNING. For pairwise learning, it is easy to verify that

$$t\text{-ACV} - t\text{-CV} = \frac{1}{p^2}\sum_{i=1}^{s} r(\mathbf{z}_i; \hat{\boldsymbol{\theta}}(\mathbf{w}')). \tag{20}$$

By Lemmas 4.2 and 4.3, when $\gamma_S = 0$, the error bound for pairwise learning can be obtained:

Theorem 4.6. *Under the same assumptions as Lemma 4.3,* $\gamma_P = 0$*,*

$$|t\text{-ACV} - t\text{-CV}| = O\left(\frac{m^2}{(p-m)^2}\right),$$

where m *is the size of the fold of dataset,* $m = p/t$*.*

Remark 7. *Particularly, when only one pair at a time is left as the validation set, the error bound of the approximate LOO is* $|ALOO - LOO| = O\left(\frac{1}{p^2}\right)$*. Previous work [19, 44] combines points into pairs and brings the pairs as a whole into the method provided by [25]. To the best of our knowledge, our research first gives the most natural formula and proof in pairwise learning.*

In this paper, we not only prove the error bound for supervised and semi-supervised learning of approximate CV, but also give the error bound for the pairwise learning of approximate CV.

5 EXTENSIONS

5.1 Non-differentiable Losses

If the derivatives $\nabla_\theta F$ and $\nabla^2_\theta F$ do not exist, we can use the smooth versions of the function to substitute the non-differentiable losses, so that the problem is solved using this approximation.

5.2 Non-convexity and Non-convergence

In Section 3, we took $\hat{\boldsymbol{\theta}}$ as the global minimum. In practice, if we get solution $\tilde{\boldsymbol{\theta}}$ from a non-convex objective or by running SGD with early stopping, $\tilde{\boldsymbol{\theta}} \neq \hat{\boldsymbol{\theta}}$.

To extend our approximate method to solve non-convex and non-convergent problems, we can form a convex quadratic approximation of the loss around $\tilde{\boldsymbol{\theta}}$, i.e.,

$$F(\mathbf{w};\boldsymbol{\theta}) \approx F(\mathbf{w};\tilde{\boldsymbol{\theta}}) + \nabla_{\boldsymbol{\theta}} F(\mathbf{w};\tilde{\boldsymbol{\theta}})(\boldsymbol{\theta}-\tilde{\boldsymbol{\theta}}) + \frac{1}{2}(\boldsymbol{\theta}-\tilde{\boldsymbol{\theta}})^{\mathrm{T}}(\mathbf{H}_{\tilde{\theta}}+\boldsymbol{\lambda}\mathbf{I})(\boldsymbol{\theta}-\tilde{\boldsymbol{\theta}}).$$

Remark 8. *From the above analysis, we know that our approximate cross-validation method can be extended to the non-convex and non-convergent situations. In this paper, we use LiSSA (Linear time Stochastic Second-Order Algorithm) [25] to add the damping term $\boldsymbol{\lambda}$. The calculation details are in Section 5.3*

Corollary 5.1. *Assume that $|F(\mathbf{w};\boldsymbol{\theta}) - \tilde{F}(\mathbf{w};\boldsymbol{\theta})| < \epsilon$, where $\tilde{F}(\mathbf{w};\boldsymbol{\theta}) = F(\mathbf{w};\tilde{\boldsymbol{\theta}}) + \nabla_{\boldsymbol{\theta}} F(\mathbf{w};\tilde{\boldsymbol{\theta}})(\boldsymbol{\theta}-\tilde{\boldsymbol{\theta}}) + \frac{1}{2}(\boldsymbol{\theta}-\tilde{\boldsymbol{\theta}})^{\mathrm{T}}(\mathbf{H}_{\tilde{\theta}}+\boldsymbol{\lambda}\mathbf{I})(\boldsymbol{\theta}-\tilde{\boldsymbol{\theta}})$ is the local convex approximation of $F(\mathbf{w};\boldsymbol{\theta})$. Under the same assumptions as Lemma 4.3, we have the conclusion that for semi-supervised learning*

$$|t\text{-}ACV - t\text{-}CV| = O\left(\frac{m^2}{(s-m)^2} + \frac{m^2}{(s-m)(p-m)} + \frac{m^2}{(p-m)^2} + \epsilon\right). \tag{21}$$

For pairwise learning, we have

$$|t\text{-}ACV - t\text{-}CV| = O\left(\frac{m^2}{(p-m)^2} + \epsilon\right), \tag{22}$$

where t-ACV is the value of the approximate method on $\tilde{F}(\mathbf{w};\boldsymbol{\theta})$ t-CV is the value of the general CV on $F(\mathbf{w};\boldsymbol{\theta})$, and m is the size of the fold of the dataset.

Remark 9. *ϵ is difficult to avoid for non-convex neural networks and is related to the nature of the model itself. Here we give an upper bound on the error in the case of convex approximation. In particular, when the size of the data is relatively large, the main source of error is basically brought by the convex approximation of the nonconvex networks, which is its own characteristics.*

6 APPLICATIONS

In this section, we will give some specific examples and technologies. Firstly we discuss specific applications from supervised learning, semi-supervised learning and pairwise learning. Since supervised learning is a special case of semi-supervised learning, we mainly focus on semi-supervised learning and pairwise learning. Then, we provide some acceleration technology used in the approximate calculation.

6.1 Semi-supervised Learning

In the linear case, the hypothesis $f_{\boldsymbol{\theta}}(\mathbf{x}) = \langle \mathbf{x}, \boldsymbol{\theta} \rangle$.

Example 1: LapRLS. The structural risk using Linear Laplacian Regularized Least Squares (LapRLS) can be wirtten as

$$\underset{\boldsymbol{\theta}\in\Theta}{argmin} \frac{1}{s}\sum_{i=1}^{s}(\boldsymbol{\theta}^{\mathrm{T}}\mathbf{x}_i - y_i))^2 + \frac{\gamma_P}{p^2}\sum_{i,j=1}^{p} W_{ij}(\boldsymbol{\theta}^{\mathrm{T}}\mathbf{x}_i - \boldsymbol{\theta}^{\mathrm{T}}\mathbf{x}_j)^2 + \gamma_I\|\boldsymbol{\theta}\|^2,$$

where ℓ_s is the square loss function, ℓ_p is the manifold-based loss function [6]. γ_S is set to 1 to simplify the calculation. $\ell_p(\mathbf{x},\mathbf{x}';\boldsymbol{\theta}) = W_{ij}(f_{\boldsymbol{\theta}}(\mathbf{x}) - f_{\boldsymbol{\theta}}(\mathbf{x}'))$, which is popular used in the semi-supervised learning, and $R(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|^2$. $\mathbf{W}$ is the affinity matrix defining the similarity between any pair of samples of $\mathcal{P}$. In this paper, we set $W_{ij} = \exp\left(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma_w\right)$ for simplicity.

Example 2: LapSVM. When we talk about SVM, the basic structural risk in Linear Laplacian Regularized Squares SVM (LapSVM) can be given as

$$\underset{\boldsymbol{\theta}\in\Theta}{argmin} \frac{1}{s}\sum_{i=1}^{s}\max(0, 1 - y\boldsymbol{\theta}^{\mathrm{T}}\mathbf{x}_i)^2 + \frac{\gamma_P}{p^2}\sum_{i,j=1}^{p} W_{ij}(\boldsymbol{\theta}^{\mathrm{T}}\mathbf{x}_i - \boldsymbol{\theta}^{\mathrm{T}}\mathbf{x}_j)^2 + \gamma_I\|\boldsymbol{\theta}\|^2,$$

where ℓ_s is the square Hinge loss function, and ℓ_p is the manifold-based loss function, γ_S is defined as 1.

Corollary 6.1. *If $\forall \mathbf{x} \in \mathcal{X}$, $y \in \mathcal{Y}$, $\|\mathbf{x}\| \leq c_1$, $|y| \leq c_2$, $\forall \boldsymbol{\theta} \in \Theta$ and $\|\boldsymbol{\theta}\| \leq c_3$. Then, for linear LapRLS (and LapSVM), the following result holds:*

$$|t\text{-}ACV - t\text{-}CV| = O\left(\frac{m^2}{(s-m)^2} + \frac{m^2}{(s-m)(p-m)} + \frac{m^2}{(p-m)^2}\right).$$

Remark 10. *We only consider the square loss, square Hinge loss, and manifold-based loss functions here. Actually, it is easy to extend our results to other loss functions, such as logistic loss, Huber loss, etc. On the other hand, we can easily derive the similar conclusion for kernel-based case. When the kernel matrix $\|\boldsymbol{\kappa}(\mathbf{x})\|$ has upper bound, we have the same result like Corollary 6.1.*

6.2 Pairwise Learning

In the case, γ_S is equal to 0, the hypothesis $f_{\boldsymbol{\theta}}(\mathbf{x})$ represents scoring of a certain data point. Meanwhile let γ_P be equal to 1.

Example 3: Common structural risk in contrastive learning can be written as

$$\underset{\boldsymbol{\theta}\in\Theta}{argmin} \frac{1}{p^2}\sum_{i,j=1}^{p}\left[(1-y)\max(\gamma_m - ||f_{\boldsymbol{\theta}}(\mathbf{x}_i) - f_{\boldsymbol{\theta}}(\mathbf{x}_j)||_2, 0)^2 + y(f_{\boldsymbol{\theta}}(\mathbf{x}_i) - f_{\boldsymbol{\theta}}(\mathbf{x}_j))^2\right] + \gamma_I\|\boldsymbol{\theta}\|^2,$$

where y indicates whether two points belong to the same class. $y = 1$ if picture i and picture j come from the same class, otherwise $y = 0$. γ_m is the threshold value for contrastive loss [12, 36].

Corollary 6.2. *If* $\forall \mathbf{x} \in \mathcal{X}$, $f_{\boldsymbol{\theta}}(\mathbf{x})$ *is the linear case, and* $\forall \boldsymbol{\theta} \in \Theta$, $\|\boldsymbol{\theta}\| \leq c_1$. *Then, for pairwise loss, the following result holds:*

$$|t\text{-}ACV - t\text{-}CV| = O\left(\frac{m^2}{(p-m)^2}\right).$$

6.3 Calculating the Inverse Hessian Efficiently

In the process of approximate calculation, computing the inverse Hessian is expensive. In this paper, we use the method provided by [1]. Unlike the previous research, they calculate the influence function for each training point with a fixed test point. We calculate our approximate value for a certain set $\mathcal{W}$ with each valid data.

First, we notice that for each test data $\mathbf{z}_i \in \mathcal{W}$ (or point-to-point $(\mathbf{z}_i, \mathbf{z}_j) \in \mathcal{W}$), we can compute

$$s_{test} = \mathbf{H}^{-1}\mathbf{g}_{\setminus \mathcal{W}}.$$

Here we use the LiSSA (Linear time Stochastic Second-Order Algorithm) to calculate s_{test} [1]. Consider $\mathbf{A}_t = \tilde{\mathbf{H}}_t^{-1} v$ as a whole. First let $v := \mathbf{g}_{\setminus \mathcal{W}}$, and initialize the inverse HVP estimation $\mathbf{A}_0 = v$. Then sample k points from the whole training data,

$$\mathbf{A}_t = v + (\mathbf{I} - \nabla^2_{\boldsymbol{\theta}} F(\mathbf{w}; \boldsymbol{\theta}))\mathbf{A}_{t-1} v.$$

Repeat the above two steps until $\mathbf{A}_t$ is stabilized and the damping term $\boldsymbol{\lambda}$ can be added in the course of the loop, where $\mathbf{A}_t$ is our final result of s_{test}. Then we can compute the approximate value for each test data with s_{test}. Using this approach, we reduce the complexity of the algorithm from $O(np^2 + p^3)$ to $O(np)$, where p is the size of the parameter.

Remark 11. *Based on the second derivative (Theorem 4.1) and the Taylor remainder [29], we present an explicit finite-sample error bound of our approximate CV as well as extensions on neural networks. Besides this, we give a method to calculate our approximate value efficiently. The framework can be used extensively in multimedia tasks such as supervised, semi-supervised and pairwise learning.*

7 EXPERIMENTS

In this section, we will do some experiments in a single machine with eight cores (Intel Xeon Silver 4210@2.20 GHz), 128 GB of memory and GeForce RTX 3090 with 24 GB graphics memory. First, we will verify the accuracy and efficiency of the approximation by the t-CV method. Then we use the approximation algorithm in this paper to filter the models with different parameters in several scenarios, such as classical semi-supervised learning and contrastive learning.

7.1 Validation

7.1.1 ACCURACY. To investigate the accuracy of our method, we do the 20-fold cross-validation [1] of the dataset and compare **Actual loss** with **Predicted loss**. To more accurately represent the role of our approximation algorithm, we calculate the difference between the loss of the validation set in the model trained with full dataset and the model with training dataset, where **Actual diff in loss** = $\ell_e(\mathbf{z}_{test}, \tilde{\boldsymbol{\theta}}(\mathbf{1})) - \ell_e(\mathbf{z}_{test}, \tilde{\boldsymbol{\theta}}(\boldsymbol{w} \setminus \mathcal{S}_i))$ and **Predicted diff in loss** = t-ACV$(\mathbf{z}_{test}) - \ell_e(\mathbf{z}_{test}, \tilde{\boldsymbol{\theta}}(\boldsymbol{w} \setminus \mathcal{S}_i))$. We use Laplacian Regularized Least Squares (LapRLS) on dataset "cpusmall" from LIBSVM Data[2] with 90% of samples randomly selected as unlabeled data.

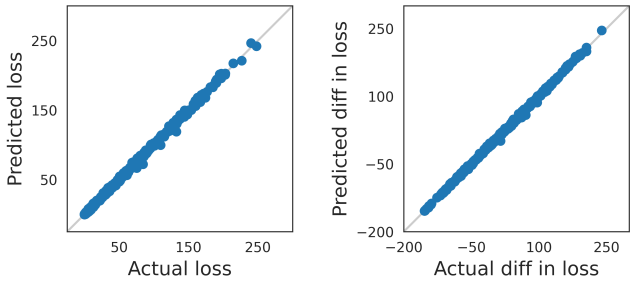

Figure 1: LEFT:Actual loss vs. Predicted loss with regression for semi-supervised learning. RIGHT: Actual diff in loss vs. Predicted diff in loss with regression for semi-supervised learning.

The result is shown in Fig. 1 (Pearson's $R = 0.99$ for both pictures). As the figure shows, the value of actual diff in loss with t-CV method is quite large. But our work can calculate their deviations accurately and efficiently. The right one in Fig. 1 shows that the finall t-ACV and t-CV is almost equal. Therefore we can use this method to efficiently evaluate models and make model selection further.

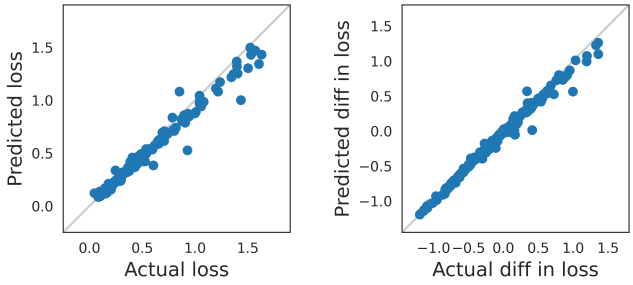

Figure 2: LEFT: Actual loss vs. Predicted loss with classification for semi-supervised learning. RIGHT: Actual diff in loss vs. Predicted diff in loss with classification for semi-supervised learning.

Fig. 2 gives another example. We use Laplacian Regularized Logistic Regression (LapRLR) on dataset "w2a" from LIBSVM Data with 90% of samples randomly selected as unlabeled data. The Pearson's $R = 0.99$ for both pictures. Our work performs well on both regression and classification problems. All we need is just one cost of trainning session.

Our approach also works well in neural network models. Next we will give the validation experiment on pairwise learning and neural networks. We use the dataset AT&T Database of Faces [3], which contains a set of face images taken between April 1992 and

[1] 20 is an appropriate number of folds that our approximation is accurate and general CV can be calculated in an accepted cost.

[2] http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[3] The database was used in the context of a face recognition project carried out in collaboration with the Speech, Vision and Robotics Group of the Cambridge University Engineering Department.

April 1994 at the lab. There are 10 different images of each of 40 distinct subjects.

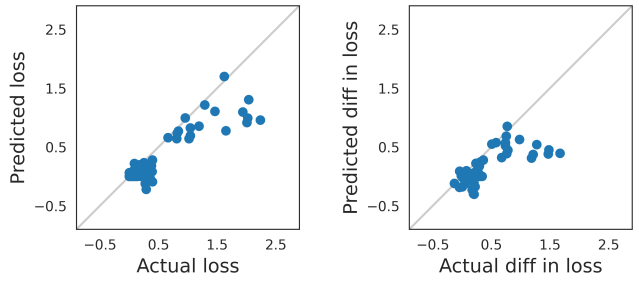

Figure 3: LEFT: Actual loss vs. Predicted loss with regression for semi-supervised learning. RIGHT: Actual diff in loss vs. Predicted diff in loss with CNN for pairwise learning.

Siamese neural network [12] is an artficial neural network that uses the same weights while working in tandem on two different input vectors to compute comparable output vectors. We use the Siamese Net to determine if two images belong to one person. The pairwise loss we use is given in **Example 3** which is a common loss function in this field. The value of the pairwise loss is small when the input pairs belong to the same class, otherwise the model returns a large loss.

We randomly choose 2 classes as the validation set and the remaining 38 classes as the training set, which is equivalent to doing a part of calculation for the 20-fold cross-validation operation. As described earlier, the loss is considered as a measure of whether two images belong to one class or not. To show the effect of our method more intuitively. Here we concentrate on exceptionally small or large losses that are typically pairs from the same class of different class.

The result is shown in Fig. 3 (Pearson's $R = 0.90$ for the left and Pearson's $R = 0.75$ for the right). We trained the model in a non-convergent, non-convex setting with a convolutional neural network. The model has not even seen the classes inside the validation set. In this difficult setting, the predicted and actual changes in loss were highly correlated.

The previous validation experiments demonstrate the accuracy of our algorithm, with excellent results in simple semi-supervised classification and regression problems. The approximation algorithm is still accurate enough even for complex neural network in pairwise learning. This is sufficient for us to use for fast model selection.

7.1.2 EFFICIENCY. The running time of t-ACV and t-CV are reported in Table 1. It is shown that t-ACV is much faster than t-CV. In particular, t-ACV is nearly t times faster than t-CV on most datasets especially when the number of folds is relatively large. The reason is very simple: although our method requires the full dataset to be used once for training, when t is small, the size of training dataset is much smaller than the whole dataset, otherwise the cost of training models using trainging dataset and whole dataset is nearly equal. Therefore roughly t-ACV is t times faster than t-CV. For large-scale datasets, the original CV methods do not work while our approximate CV does.

Table 1: Running time. Our methods: t-ACV, compared methods: t-CV, $t = 5, 10, 20$

datasets	5-CV	5-ACV	10-CV	10-ACV	20-CV	20-ACV
abalone	20.65	5.59	52.96	6.28	109.81	6.72
cpusmall	97.25	21.41	207.64	22.33	422.21	23.92
YearPrediction	\	4623.78	\	4823.56	\	5023.24
svmguide3	8.76	5.43	35.85	5.86	91.48	6.08
w2a	220.71	51.54	486.57	52.76	1003.15	53.84
a9a	\	1699.53	\	2121.24	\	2721.23
AT&T	\	99.40	\	102.22	\	109.46

Table 2: Hyperparameter selection. Our methods: t-ACV, compared methods: 20-CV

datasets	20-CV γ_S	20-ACV γ_S	20-CV γ_P	20-ACV γ_P	20-CV γ_I	20-ACV γ_I
abalone	1	1	0.1	0.1	0.1	0.1
cpusmall	1	1	1	1	0.1	0.1
YearPrediction	\	1	\	0.1	\	1
svmguide3	1	1	0.1	0.1	0.01	0.01
w2a	1	1	0.1	0.1	0.1	0.1
a9a	\	1	\	1	\	0.1
AT&T	\	0	\	1	\	0.1

7.2 Model Selection

Table 2 gives the results of hyperparameter selection for different datasets, where the datasets "abalone", "cpusmall" and "YearPrediction" are used for the regression, "svmguide3", "w2a", "a9a" for classification and "AT&T" for pairwise learning. All the parameters are $\gamma \in \{10^i, i = -3, -2, -1, 0, 1, 2, 3\}$. To simplify our calculation, we set $\gamma_S = 1$ for semi-supervised learning, and $\gamma_S = 0, \gamma_P = 1$ for pairwise learning. For small-scale datasets, we compare the orginal CV methods with our approximate CV. But the orginal methods do not work for large-scale datasets, so that we use "\" to denote the inability to calculate. At this point our approach can also select the proper hyperparameters.

The results shows that our unified framework works well in different scenarios of model selection such as supervised learning, semi-supervised learning and pairwise learning, which is useful for multimedia tasks with a large number of unsupervised samples and contrast learning applications. The complex experiment on the images also proves the effectiveness of our method, which is also relevant in neural networks.

8 CONCLUSION

In this paper, we present a unified framework to approximate the CV which can be used in supervised, semi-supervised, and pairwise learning, and provide a theoretical guarantee for its performance. The proposed approximate CV requires training on the full dataset only once so that they can significantly improve the efficiency. Experimental results on several datasets demonstrate that our approximate CV is much more efficient with accuracy guaranteed and has no statistical discrepancy compared with the original CV, which can be widely used in multimedia tasks. Finally we give specific examples of model selection in semi-supervised learning and pairwise learning using neural network.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China NO. 62076234, the Beijing Outstanding Young Scientist Program NO. BJJWZYJH012019100020098, China Unicom Innovation Ecological Cooperation Plan, the Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China, and the Public Computing Cloud of Renmin University of China.

REFERENCES

[1] Naman Agarwal, Brian Bullins, and Elad Hazan. 2016. Second-order stochastic optimization in linear time. *stat* 1050 (2016), 15.

[2] Ahmed Alaa and Mihaela Van Der Schaar. 2019. Validating causal inference models via influence functions. In *Proceedings of the 36th International Conference on Machine Learning (ICML*.

[3] Senjian An, Wanquan Liu, and Svetha Venkatesh. 2007. Fast cross-validation algorithms for least squares support vector machine and kernel ridge regression. *Pattern Recognition* 40, 8 (2007), 2154–2162.

[4] Elnaz Barshan, Marc-Etienne Brunet, and Gintare Karolina Dziugaite. 2020. Relatif: Identifying explanatory training samples via relative influence. In *Proceedings of the 23th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

[5] Ahmad Beirami, Meisam Razaviyayn, Shahin Shahrampour, and Vahid Tarokh. 2017. On optimal generalizability in parametric learning. In *Advances in Neural Information Processing Systems (NIPS)*.

[6] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. 2006. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research* 7 (2006), 2399–2434.

[7] Yoshua Bengio and Yves Grandvalet. 2004. No unbiased estimator of the variance of k-fold cross-validation. *Journal of Machine Learning Research* 5 (2004), 1089–1105.

[8] Gavin C Cawley. 2006. Leave-one-out cross-validation based model selection criteria for weighted LS-SVMs. In *The 2006 IEEE international joint conference on neural network proceedings*.

[9] Gavin C Cawley and Nicola LC Talbot. 2004. Fast leave-one-out cross-validation of sparse least-squares support vector machines. *Neural Networks* 17, 10 (2004), 1467–1475.

[10] Gavin C. Cawley and Nicola L. C. Talbot. 2007. Preventing over-fitting during model selection via Bayesian regularisation of the hyper-parameters. *Journal of Machine Learning Research* 8 (2007), 841–861.

[11] Weiyu Cheng, Yanyan Shen, Linpeng Huang, and Yanmin Zhu. 2019. Incorporating interpretability into latent factor models via fast influence analysis. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (SIGKDD)*. 885–893.

[12] Sumit Chopra, Raia Hadsell, and Yann LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*.

[13] Michiel Debruyne. 2007. *Robustness of censored depth quantiles, PCA and kernel based regression, with new tools for model selection*. Ph.D. Dissertation. Katholieke Universiteit Leuven.

[14] Michiel Debruyne, Mia Hubert, and Johan A.K. Suykens. 2008. Model selection in kernel based regression using the influence function. *Journal of Machine Learning Research* 9 (2008), 2377–2400.

[15] Minghong Fang, Neil Zhenqiang Gong, and Jia Liu. 2020. Influence function based data poisoning attacks to top-n recommender systems. In *Proceedings of The Web Conference 2020 (WWW)*.

[16] Ryan Giordano, Michael I Jordan, and Tamara Broderick. 2019. A higher-order Swiss Army infinitesimal jackknife. *arXiv preprint arXiv:1907.12116* (2019).

[17] Ryan Giordano, Will Stephenson, Runjing Liu, Michael I Jordan, and Tamara Broderick. 2019. A Swiss Army Infinitesimal Jackknife. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*.

[18] Dany Haddad and Joydeep Ghosh. 2019. Learning more from less: Towards strengthening weak supervision for ad-hoc retrieval. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*.

[19] Hojae Han, Seung-won Hwang, Young-In Song, and Siyeon Kim. 2020. Training Data Optimization for Pairwise Learning to Rank. In *Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval (SIGIR)*.

[20] Xiaochuang Han, Byron C Wallace, and Yulia Tsvetkov. 2020. Explaining black box predictions and unveiling data artifacts through influence functions. *arXiv preprint arXiv:2005.06676* (2020).

[21] Jay Heo, Junhyeon Park, Hyewon Jeong, Kwang Joon Kim, Juho Lee, Eunho Yang, and Sung Ju Hwang. 2020. Cost-effective Interactive Attention Learning with Neural Attention Processes. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*.

[22] Zachary Izzo, Mary Anne Smart, Kamalika Chaudhuri, and James Zou. 2021. Approximate Data Deletion from Machine Learning Models. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

[23] Julie Josse and François Husson. 2012. Selecting the number of components in principal component analysis using cross-validation approximations. *Computational Statistics and Data Analysis* 56, 6 (2012), 1869–1879.

[24] Rajiv Khanna, Been Kim, Joydeep Ghosh, and Sanmi Koyejo. 2019. Interpreting black box predictions using fisher kernels. In *The 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*.

[25] PangWei Koh and Percy Liang. 2017. Understanding Black-box Predictions via Influence Functions. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*.

[26] Pang Wei Koh, Kai-Siang Ang, Hubert HK Teo, and Percy Liang. 2019. On the accuracy of influence functions for measuring group effects. *arXiv preprint arXiv:1905.13289* (2019).

[27] R Kohavi. 1995. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Conference on Artificial Intelligence (IJCAI)*.

[28] Ravi Kumar, Daniel Lokshtanov, Sergei Vassilvitskii, and Andrea Vattani. 2013. Near-optimal bounds for cross-validation via loss stability. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*.

[29] Seppo Linnainmaa. 1976. Taylor expansion of the accumulated rounding error. *BIT Numerical Mathematics* 16, 2 (1976), 146–160.

[30] Yong Liu, Shali Jiang, and Shizhong Liao. 2014. Efficient approximation of cross-validation for kernel methods using Bouligand influence function. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*.

[31] Yong Liu and Shizhong Liao. 2014. Preventing over-fitting of cross-validation with kernel stability. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 290–305.

[32] Yong Liu and Shizhong Liao. 2017. Granularity selection for cross-validation of SVM. *Information Sciences* 378 (2017), 475–483.

[33] Yong Liu, Shizhong Liao, Shali Jiang, Lizhong Ding, Hailun Lin, and Weiping Wang. 2020. Fast Cross-Validation for Kernel-based Algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42, 5 (2020), 1083–1096.

[34] Yong Liu, Hailun Lin, Li-Zhong Ding, Weiping Wan, and Shizhong Liao. 2018. Fast Cross-Validation.. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*.

[35] Måns Magnusson, Michael Andersen, Johan Jonasson, and Aki Vehtari. 2019. Bayesian leave-one-out cross-validation for large data. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*.

[36] Mohammad Norouzi, David J Fleet, and Russ R Salakhutdinov. 2012. Hamming distance metric learning. In *Advances in Neural Information Processing Systems (NIPS)*.

[37] Tapio Pahikkala, Jorma Boberg, and Tapio Salakoski. 2006. Fast n-fold cross-validation for regularized least-squares. In *Proceedings of the 9th Scandinavian conference on artificial intelligence (SCAI)*, Vol. 83. Espoo, 90.

[38] Matthias W Seeger. 2008. Cross-validation optimization for large scale structured classification kernel methods. *The Journal of Machine Learning Research* 9 (2008), 1147–1178.

[39] Xiaoting Shao, Arseny Skryagin, P Schramowski, W Stammer, and Kristian Kersting. 2021. Right for Better Reasons: Training Differentiable Models by Constraining their Influence Function. In *Proceedings of 35th AAAI Conference on Artificial Intelligence (AAAI)*.

[40] Boris Sharchilev, Yury Ustinovskiy, Pavel Serdyukov, and Maarten Rijke. 2018. Finding influential training samples for gradient boosted decision trees. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*.

[41] Daniel Ting and Eric Brochu. 2018. Optimal subsampling with influence functions. In *Advances in Neural Information Processing Systems (NIPS)*.

[42] Hao Wang, Berk Ustun, and Flavio Calmon. 2019. Repairing without retraining: Avoiding disparate impact with counterfactual distributions. In *In Proceedings of the 36th International Conference on Machine Learning (ICML)*.

[43] Tianyang Wang, Jun Huan, and Bo Li. 2018. Data dropout: Optimizing training data for convolutional neural networks. In *2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)*.

[44] Zifeng Wang, Hong Zhu, Zhenhua Dong, Xiuqiang He, and Shao-Lun Huang. 2020. Less is better: Unweighted data subsampling via influence function. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI)*.

[45] Ashia Wilson, Maximilian Kasy, and Lester Mackey. 2020. Approximate cross-validation: Guarantees for model assessment and selection. In *Proceedings of the 23th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

[46] Hengtong Zhang, Yaliang Li, Bolin Ding, and Jing Gao. 2020. Practical Data Poisoning Attack against Next-Item Recommendation. In *Proceedings of The Web Conference 2020 (WWW)*.